(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 December 2000 (21.12.2000)

PCT

(10) International Publication Number
WO 00/77985 A1

(51) International Patent Classification[7]: H04L 12/56

(21) International Application Number: PCT/US00/11081

(22) International Filing Date: 25 April 2000 (25.04.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/329,899 10 June 1999 (10.06.1999) US

(71) Applicant: ENTERA, INC. [US/US]; 40971 Encyclopedia Circle, Fremont, CA 94538 (US).

(72) Inventors: GEAGAN, John, B., III; 1690 Ambergrove Drive, San Jose, CA 95131 (US). KELLNER, Michael, D.; 2506 Kingwood Drive, Santa Clara, CA 95051 (US). PERIYANNAN, Alagu, S.; 34113 Finnigan Terrace, Fremont, CA 94555 (US).

(74) Agents: FAHMI, Tarek, N. et al.; Blakely, Sokoloff, Taylor & Zafman LLP, 7th floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025 (US).

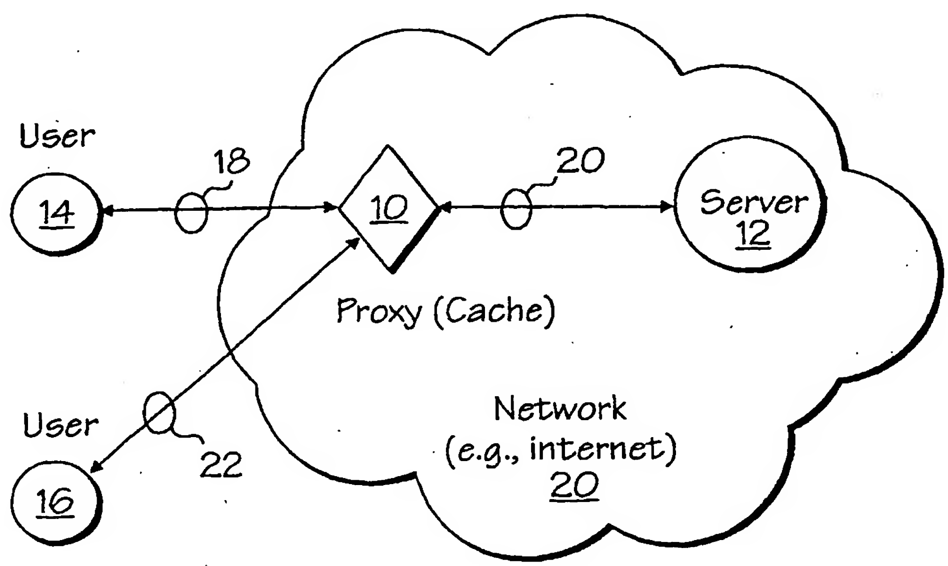
(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *With international search report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHOD AND APPARATUS FOR NETWORK TRAFFIC SMOOTHING



(57) Abstract: A burst of information included in a data stream received from a content source across one or more computer networks (e.g., according to RTP) is smoothed at a proxy disposed between the content source and one or more content consumers so as to provide one or more output data streams to the one or more content consumers that reproduce the information included in the burst over a time interval longer than that in which it was received at the proxy. The proxy may be disposed at the last media link between the content source and at least one of the content consumers.

WO 00/77985 A1

METHOD AND APPARATUS FOR NETWORK TRAFFIC SMOOTHING

FIELD OF THE INVENTION

The present invention relates to a scheme for delivering streaming content via a computer network or a network of networks such as the Internet.

BACKGROUND

The Internet is a vast and expanding network of networks of computers and other devices linked together by various telecommunications media, enabling all these computers and other devices to exchange and share data. Sites on the Internet provide information about a myriad of corporations and products, as well as educational, research and entertainment information and services. An estimated 30 million people worldwide use the Internet today, with 100 million predicted to be on the "net" in a matter of years.

A computer or resource that is attached to the Internet is often referred to as a "host." Examples of such resources include conventional computer systems that are made up of one or more processors, associated memory (typically volatile and non-volatile) and other storage devices and peripherals that allow for connection to the Internet or other networks (e.g., modems, network interfaces and the like). In most cases, the hosting resource may be embodied as hardware and/or software components of a server or other computer system that includes an interface, which allows for some dialog with users thereof. Generally, such a server will be accessed through the Internet (e.g., via Web browsers such as Netscape's Navigator™ and Communicator™ and Microsoft's Internet Explorer™) in the conventional fashion.

Briefly, if an Internet user desires to establish a connection with a host (e.g., to view a Web page located thereat), the user might enter into a Web browser program the URL (or Web address) corresponding to that host. One example of such a URL is "http://www.domain.com". In this example, the first element of the URL is a transfer protocol (most commonly, "http" standing for hypertext transfer protocol, but others include "mailto" for electronic mail, "ftp" for file transfer protocol, and "nntp" for network news transfer

protocol). The remaining elements of this URL (in this case, "www" standing for World Wide Web--the Internet's graphical user interface--and "domain.com") are an alias for the "fully qualified domain name" of the host.

Each fully qualified domain name, in its most generic form, includes three elements. Taking "computer.host.com" as an example, the three elements are the hostname ("computer"), a domain name ("host") and a top-level domain ("com"). Further, each fully qualified domain name is unique throughout the Internet and corresponds to a numerical Internet protocol (IP) address. IP addresses facilitate communications between hosts and clients in the same way that physical addresses (e.g., 123 Main Street, Anytown, Anycity) facilitate correspondence by mail. Each IP address is made up of four groups of numbers separated by decimals. Thus, in the case of the hypothetical host "computer.domain.com", the corresponding IP address might be 123.456.78.91. A given host looks up the IP addresses of other hosts on the Internet through a system known as domain name service.

Thus, once a URL is entered into a browser, the corresponding IP address is looked up in a process facilitated by a top-level server. In other words, all queries for addresses are routed to certain computers, the so-called top-level servers. The top-level server matches the domain name to an IP address of a domain name server capable of directing the inquiry to the computer hosting the sought after Web page (or other content) by matching an alphanumeric name such as www.domain.com with its numeric IP address.

In addition to Web pages and the like, more and more Internet users are accessing multimedia content (e.g., files that include high quality graphical images, movies and/or sound). This creates difficulties because such files are usually quite large while the bandwidth available through the Internet is limited. Thus, in order to make multimedia files usable, streaming is often employed.

With conventional files (e.g., data files), clients (e.g., Web browsers) completely download the requested content before viewing it. This technique works well for relatively small files, but often suffers from unacceptable (from the point of view of the user) delays when large multimedia files are involved. Streaming is the term given to a technique wherein a client downloads a portion of a file, decompresses (if necessary) that portion, and starts

playing the contents thereof (e.g., audio and/or video) before the rest of the file arrives. A buffer of information is built up before playback starts, so as to prevent underflows if the remaining data is delayed during transmission. Furthermore, subsequent portions of the multimedia file are downloaded during playback to keep the buffer relatively full. This technique thus accommodates the downloading and playing of large multimedia files without incurring lengthy delays before the content is available for viewing.

Multimedia files are often transported over the Internet using special transport protocols. For example, the real-time transport protocol (RTP) provides delivery service for multimedia applications and also provides means for multimedia applications to work over networks. RTP does not, however, provide guaranteed or in-sequence delivery (and hence it is referred to as an unreliable transport protocol), but does provide a packet sequence number that can be used to detect missing packets and to reconstruct an original transmission sequence.

RTP usually carries data in the form of packets, using the user datagram protocol (UDP) as the delivery mechanism. UDP provides a "wrapper" around data packets, with the wrapper providing for multiplexing and demultiplexing as well as error checking services. Essentially, a UDP packet is made up of a UDP header and UDP data encapsulated as the data portion of an IP packet. The IP packet itself includes an IP header (which includes the address information discussed above) as well as the user data (i.e. the multimedia content of interest) as a payload.

In some cases, RTP is used with other protocols, such as the transmission control protocol (TCP). Unlike UDP, TCP provides a reliable, error-free, full-duplex channel between two computers. TCP uses IP to transfer data, but provides mechanisms to take care of lost or duplicated IP datagrams (i.e., packets) and to ensure proper sequencing thereof. Thus, TCP provides reliable end-to-end transport, ensuring that what is received is an exact duplicate of what is transmitted.

When an application starts an RTP session, a second port for communication according to the real time control protocol (RTCP) is opened. RTCP works in conjunction with RTP to provide flow control and congestion control services. The idea is that the

exchange of RTCP packets between a client and server can be used to adjust the rate of transmission of the RTP packets, etc.

Associated with RTP is the real time streaming protocol (RTSP). RTSP is a client-server multimedia presentation control protocol that control functionality such as content type, interactive stream control, error mitigation, bandwidth negotiation, multicast, live broadcasting and monitoring. Just as HTTP transports HTML (hypertext markup language--an instruction set that allows an HTTP client to render a desired image, etc.), RTSP handles data. The difference is that while HTTP clients always make requests and HTTP servers always service those requests, RTSP is bi-directional, with both servers and clients making requests and servicing them. RTSP accomplishes data transfer using TCP or UDP.

Thus, RTSP is a generally a TCP connection over which commands are sent and responses are returned. Clients negotiate data channels with servers via SETUP commands. These channels typically specify that data will be sent as RTP/RTCP "packets", but the transport type may be specified and/or modified as part of the negotiation. Further, the negotiations may determine whether the packets will be sent over TCP (e.g., as binary packet data imbedded in an RTSP command stream via an escape sequence) or UDP (e.g., as true packets).

The RTP portion of a channel contains actual media data for single stream flows (e.g., compressed audio data). In contrast, an RTCP portion of a channel (which typically is assigned one UDP port number or TCP channel number larger than the RTP port number or channel-- for example, UDP port 6970 for RTP and 6971 for RTCP) usually contains clock-synchronization data and client-server control/status messages. As indicated above, RTP data typically flows in one direction, from the server to the client. RTCP packets are typically sent in both directions, for example as client status report messages and server status report messages.

The following dialog illustrates (from a client's point of view) a portion of an RTSP session wherein a media description is retrieved and the single audio stream specified thereby is played. Those of ordinary skill in the art will recognize that this is a “live” stream as opposed to prerecorded content, as is evident from the absence of a “range” tag in a session

description protocol (SDP). In the initial client-to-server communication, a request to describe the stream located at a particular Web address (rtsp://qt.macroradio.net/gogaga) is sent, along with suggested connection parameters, such as a bandwidth:

```
Send data (130 bytes).
<00000000< DESCRIBE rtsp://qt.macroradio.net/gogaga RTSP/1.0
<00000033< CSeq: 1
<0000003C< Accept: application/sdp
<00000055< Bandwidth: 112000
<00000068< User-Agent: QTS/1.0b22
<00000080<
```

The server responds with a description of the stream as part of an SDP:

```
Receive data (566 bytes).
>00000000> RTSP/1.0 200 OK
>00000011> Server: QTSS/v65
>00000023> Cseq: 1
>0000002C> Content-Type: application/sdp
>0000004B> Content-Base: rtsp://qt.macroradio.net/gogaga/
>0000007B> Content-length: 420
>00000090>
>00000092> v=0
>00000097> o= 3134316689 3134906918 IN IP4 192.231.139.83
>000000C7> s=GoGaGa Brand Radio
>000000DD> i=«A9»1999 - All rights reserved -
>000000FF> u=http://www.macroradio.net
>0000011C> a=x-qt-text-nam:GoGaGa Brand Radio
>00000140> a=x-qt-text-cpy:«A9»1999 - All rights reserved
>0000016D> a=x-qt-text-cpy:(c) 1999 ERC: The Eclectic Radio Company,
>000001A7> LLC.
>000001AD> t=3134316689 3134320289
>000001C6> c=IN IP4 0.0.0.0
>000001D8> a=control:*
>000001E5> m=audio 0 RTP/AVP 97
>000001FB> a=rtpmap:97 X-QT
>0000020D> a=x-bufferdelay:10
>00000221> a=control:trackID=1
```

The client then defines conditions for a playback, using a specified port:

```
Send data (143 bytes).
<00000082< SETUP rtsp://qt.macroradio.net/gogaga/trackID=1 RTSP/1.0
<000000BC< CSeq: 2
<000000C5< Transport: RTP/AVP;unicast;client_port=6970-6971
<000000F7< User-Agent: QTS/1.0b22
<0000010F<
```

The server responds with the port address from which it will play:

```
Receive data (165 bytes).
>00000236> RTSP/1.0 200 OK
>00000247> Server: QTSS/v65
>00000259> Cseq: 2
>00000262> Session: 1411920655;timeout=60
>00000282> Transport: rtp/avp;source=192.231.139.182;server_port=2000-2001;
>000002C2> client_port=6970-6971
>000002D9>
```

The client then initiates play back using the specified port addresses:

```
Send data (125 bytes).
<00000111< PLAY rtsp://qt.macroradio.net/gogaga RTSP/1.0
<00000140< CSeq: 3
<00000149< Range: npt=0.000000-
<0000015F< Session: 1411920655
<00000174< User-Agent: QTS/1.0b22
<0000018C<
```

In response, the server begins playing the selected stream:

```
Receive data (92 bytes).
>000002DB> RTSP/1.0 200 OK
>000002EC> Server: QTSS/v65
>000002FE> Cseq: 3
>00000307> Session: 1411920655
>0000031C> RTP-Info: url=trackID=1
>00000335>
```

At this point, RTP media data and RTCP control packets will start flowing between the specified UDP ports, i.e., RTP data from server port 2000 to client port 6970, and RTCP packets between server port 2001 and client port 6971. Note that the server's address 192.231.139.182 can be seen in the SETUP response above. An example of an RTCP packet transmitted from the client (port 6971) to the server (port 2001) is shown below:

```
Send packet data to port 1 (84 bytes).
<00000000< 80 C9 00 01  00 00 62 76  81 CA 00 12  00 00 62 76  ......bv......bv
<00000010< 01 0D 51 54  53 20 38 30  35 32 31 38  36 39 33 02  ..QTS 805218693.
<00000020< 13 51 75 69  63 6B 54 69  6D 65 20 53  74 72 65 61  .QuickTime Strea
<00000030< 6D 69 6E 67  06 1A 51 75  69 63 6B 54  69 6D 65 20  ming..QuickTime
<00000040< 53 74 72 65  61 6D 69 6E  67 2F 31 2E  30 62 32 32  Streaming/1.0b22
<00000050< 00 00 00 D2                                         ....
```

If one were to decode the data, it may translate to "client received 5% loss and 123456 bytes and client "name" is 'QuickTime Streaming' client".

An RTCP packet sent from the server (port 2001) to the client (port 6971) may resemble the following:

```
Receive packet data from 192.231.139.182:2001 (108 bytes).
>00000000> 80 C8 00 06  00 00 45 38  BA F0 68 00  42 1F 8E 44     ......E8..h.B..D
>00000010> 3D 25 45 EB  00 E7 A3 3F  BB 8C 3A 78  81 CA 00 13     =%E....?..:x....
>00000020> 00 00 45 38  01 18 51 54  53 40 6C 6F  63 61 6C 68     ..E8..QTS@localh
>00000030> 6F 73 74 20  31 35 36 31  36 31 35 34  36 39 02 13     ost 1561615469..
>00000040> 51 75 69 63  6B 54 69 6D  65 20 53 74  72 65 61 6D     QuickTime Stream
>00000050> 69 6E 67 06  13 51 75 69  63 6B 54 69  6D 65 20 53     ing..QuickTime S
>00000060> 74 72 65 61  6D 69 6E 67  00 FF 00 40                  treaming...@
```

Once decoded, this information may state, "RTP time t means universal (wall clock) time y and server "name" is "QuickTime Streaming server". This is essentially a reference to an absolute reference time source used by the server.

The actual RTP media packets transmitted from the server (port 2000) to the client (port 6970) may resemble the following:

```
Receive packet data from 192.231.139.182:2000 (200 bytes).
>00000000> 80 E1 BC 73  3D 21 F8 1B  00 00 45 38  14 00 80 01     ...s=!....E8....
>00000010> 02 B6 4B 19  09 14 80 9E  5D 26 35 24  88 64 2A 20     ..K.....]&5$.d*
>00000020> D0 C2 98 16  92 55 41 9C  82 46 16 35  9D A8 D7 27     .....UA..F.5...'
>00000030> 13 1A B7 37  D6 E4 05 5B  40 AF E7 11  D3 84 9C B8     ...7...[@.......
>00000040> 45 8D 51 01  F1 A4 C5 97  0B 58 88 2A  4A D1 C4 13     E.Q......X.*J...
>00000050> FC 8C 58 A5  46 8A A2 3B  63 66 6F 23  2F 38 1B 61     ..X.F..;cfo#/8.a
>00000060> 0B 15 2A D3  49 22 C9 98  C8 0F 16 40  1A 53 9D A8     ..*.I".....@.S..
>00000070> 79 F1 CE EE  C6 19 B1 26  C5 A8 CB 4D  4B 3B F3 73     y......&...MK;.s
>00000080> 4C 6A 33 5F  D3 5F 2C 46  60 84 C0 08  14 14 26 EC     Lj3_._,F`.....&.
>00000090> 5E DF 49 49  48 B5 B3 02  5F 88 F5 EC  29 10 AB 72     ^.IIH..._...)..r
>000000A0> A6 D8 3E D4  9A D2 14 2A  6F 86 AD 22  9E 0B 4C 50     ..>....*o.."..LP
>000000B0> 5C BC 0B 88  6D 13 0C 34  3C 44 CB 92  BB 6B 1B 18     \...m..4<D...k..
>000000C0> 51 1C 7D 12  01 00 00 00                               Q.}.....
```

Finally, upon conclusion of the playback or at some other point, the client may decide to quit, so an instruction is passed to server over RTSP (the TCP connection is still open during the playback) to stop everything on this "session":

```
Send data (107 bytes).
<0000018E< TEARDOWN rtsp://qt.macroradio.net/gogaga RTSP/1.0
<000001C1< CSeq: 4
<000001CA< Session: 1411920655
<000001DF< User-Agent: QTS/1.0b22
<000001F7<
```

Multimedia files such as videos and the like are often stored in a packetized form consisting of keyframes and difference frames. Different multimedia protocols use different terminology to describe these elements, but essentially in multimedia terminology, a "frame" generally represents a complete video picture. A keyframe is usually thought of as an effect that has been stored in memory, similar to a snapshot photograph. Individual keyframes can be strung together to create an overall keyframe effect, which is similar to an animation. Difference frames include information that varies from that included in prior keyframes or other difference frames. Thus, when the frames are replayed, the viewer (i.e., the client application that allows a user to view the video) transitions from one frame to the next by applying the difference frames.

The use of such keyframes and difference frames poses a potential problem for the viewing of streaming content over public networks or networks of networks such as the Internet. For example, during an RTP/RTSP streaming media session, if a client requests data from a point other than the beginning of a stored movie or other multimedia file, that point may not correspond to a keyframe. Hence, the server storing the requested file may, in order to be able to allow the client to view the requested information, have to transmit a significant amount of data (i.e., the preceding keyframe and one or more intervening difference frames). That is, in order to provide enough information for the client to render the requested image, etc., it is not sufficient for the server simply to begin transmitting data from the point indicated by the client. Sufficient preceding information must also be transmitted in order to provide the client with instructions and data sufficient to allow the client to reconstruct the requested images, etc.

Typically, this information is not transmitted in the typical fashion in which the multimedia file is ordinarily played out. Instead, it is transmitted as a burst of data, which may very well overwhelm the requesting client (i.e., by overflowing the client's receive buffer) and/or the bandwidth of the connection between the server and the client. Moreover, usually such bursts are transmitted via UDP, meaning that any packets that are not properly delivered to the client will be dropped.

Figure 1 illustrates this problem graphically. Assume that a user has requested download of a streaming multimedia file from a point several frames into the file. This may correspond to a user requesting play back of a particular portion of a movie or other streaming content. As shown near the origin point of the graph, the server responds to the download request by transmitting a burst of information that is greater than the maximum bandwidth (m) of the connection to the client. This bandwidth limit may represent the bandwidth of the client's receive buffer, the media transporting the requested information, any or all intervening switches and routers, or in the general case a combination of these factors. After the initial burst, the streaming content is provided at a fairly uniform rate (perhaps with some variation due to network traffic congestion, etc.) that is easily accommodated within the bandwidth threshold m. However, the initial information burst will cause packet loss.

The net result of the above scenario is a poor user experience and possibly increased delay in beginning play back while the client attempts to reconstruct the multimedia file (e.g., possibly by requesting retransmissions of dropped packets or perhaps by initiating play back from the point of the next keyframe, etc.). What is needed therefore is a scheme to deal with these network traffic conditions.

SUMMARY OF THE INVENTION

In one embodiment, a burst of information included in a data stream received from a content source across one or more computer networks (e.g., according to RTP) is smoothed at a proxydisposed between the content source and one or more content consumers so as to provide one or more output data streams to the one or more content consumers that reproduce the information included in the burst over a time interval longer than that in which it was received at the proxy. The proxymay be disposed at the last media link between the content source and at least one of the content consumers.

Other features and advantages of the present invention will be apparent from the following discussion.

Brief Description of the Drawings

The present invention is illustrated by way of example, and not limitation, in the figures of the accompanying drawings in which like reference numerals refer to similar elements and in which:

Figure 1 illustrates a situation in which a content source bursts requested streaming content to a content consumer across the Internet;

Figure 2 illustrates the smoothing of the information burst shown in Figure 1 through the use of a cache disposed in a communication path between a content source and a content consumer in accordance with an embodiment of the present scheme;

Figure 3 illustrates a cache disposed in a communication path between a content source and a content consumer in accordance with an embodiment of the present scheme; and

Figure 4 is a functional depiction of a cache configured to perform traffic smoothing operations in accordance with an embodiment of the present scheme.

Detailed Description

Disclosed herein is a scheme for smoothing bursts of streaming content downloaded over a public network or network of networks such as the Internet. In essence, a proxy (which, in some cases may be associated with a client or a content source such as a server and which may be transparent or explicit) is introduced between a content source (e.g., a server) and one or more clients (e.g., Web browsers, or plug-ins therefor, configured to play streaming content or other multimedia viewers), possibly at a location that is close (e.g., physically or logically) to the clients. Herein, the term proxy is meant to describe and/or refer to a device that resides logically between a client and a server, or other content source, and that processes information flowing there between in some manner. Proxies may be physically co-located with clients and/or servers and/or may be stand-alone devices. A data stream (transported according to an unreliable transport protocol such as RTP) from the source is received at the proxyand from there is routed to the requesting client(s). En route, the stream can be stored and any information bursts (e.g., due to the content source providing information needed to allow the client(s) to view the requested material) in the received

streams can be smoothed over time to provide the requested play backs at bandwidths suitable for the client(s).

Although discussed with reference to certain illustrated embodiments, upon review of this specification, those of ordinary skill in the art will recognize that the present scheme may find application in a variety of systems, perhaps with one or more minor variations. Therefore, in the following description the illustrated embodiments should be regarded as exemplary only and should not be deemed to be limiting in scope. Further, it should be kept in mind that some portions of the detailed description that follows are presented in terms of algorithms and symbolic representations (e.g., through the use of graphs, etc.) of operations on data within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the computer science arts to most effectively convey the substance of their work to others skilled in the art.

An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers or the like. It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities.

Moreover, unless specifically stated otherwise, it will be appreciated that throughout the description of the present scheme, use of terms such as "processing", "computing, "calculating", "determining", "displaying", "rendering" or the like, refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or

display devices. Again, these are the terms and descriptions commonly used by and among practitioners of ordinary skill in the relevant arts.

As described above, the present scheme introduces a proxybetween the content source (e.g., a server or another proxy) and various users (e.g., client applications such as Web browsers that may operate in conjunction with media player plug-ins and/or helper applications), as shown in **Figure 2**. Assuming the connection between the proxyand the content source (termed the origin server in the illustration) has a bandwidth sufficient to accommodate the information burst provided by the server at the start of the requested playback, that burst is stored at the proxy. Later, the proxyplays out the data transferred in the burst to the client, however, this is done over a longer time interval that that in which the information was originally received at the proxy. In essence, the proxysmoothes out the burst of information over time when playing out the data to the client. As shown in the top portion of the graph, this smoothing effect allows the client to download all of the information included in the burst, without exceeding the bandwidth threshold (m).

In the example shown in **Figure 2,** the proxyintroduces a one time unit delay from the start of the server's original transmission of the information burst into the stream played out to the client. This allows time for the proxyto store the bulk of the information being transmitted by the server. Then, the information received during the one time unit period is played out to the client over a period of several time units. This "time stretching" thus provides a longer period in which the large amount of data originally burst by the server can be received by the client. Over time, as the server returns to a smoother information transfer, the proxybegins to transmit this data to the client over a more equal time period, but still with some delay. This allows the client to receive data at the expected rate according to the transport protocol being used, and also allows the proxyto maintain a buffer of information so as to help prevent underflows and overflows at the client.

In one embodiment, the proxyintroduces a constant one unit of time delay from all transmissions received from the server. At the outset (i.e., to accommodate the burst from the server), this delay is augmented with a time stretch so as not to overwhelm the receiving

client. Thus, the proxymay play out information received during a unit's worth of time from the server over a time t_{play}, where:

$$t_{play} = 1 + (1 - \tfrac{m}{n}), m = 0,1,2...n$$

In this expression, m may increment by intervals greater than 1 and/or be dependent upon the amount of information received in a previous and/or current time interval.. In other words, the amount of data to be added to the data stream can be incremented to just the limit of the maximum available bandwidth. Also, n is variable and represents the number of time segments until the streams that are being received from the server are in 1:1 correspondence with the streams that are being transmitted from the proxy.

In other words, for the information received from the server during the first unit of time, the proxywill play out that same information over a period $t_{play} = 1 + (1 - 1/n)$. Then, for information received over the next unit of time, the play out period becomes $1 + (1 - 2/n)$. This continues for "n" time periods, until $t_{play} = 1 + (1 - n/n) = 1$ -- the equivalent time interval to that in which the information was received from the server. At this point, the information received from the server is being delayed in its play out to the client, but no further time stretch is being introduced. This effect is shown in Figure 2, where as time passes, the time stretch introduced by the proxyis gradually eliminated. In other cases, the time intervals may be manipulated to accommodate bursts that occur after an initial burst, e.g., due to network delays, etc.

Figure 3 shows how a proxy10 may be introduced in a communication path between content source (e.g., server 12) and content users 14 and 16. In one embodiment, the proxy10 is introduced as close to the last physical media link to the users as possible. Thus, the proxy 10 may be situated at the point a user's dial up Internet connection is terminated (e.g., as deployed by an Internet service provider as part of a modem bank). Now, when a user (or more particularly a client application being run on a computer platform operated by a user) connects to server 12, the connection actually passes through proxy10. Thus, user 14 has a connection (e.g., a dial up connection) 18 to proxy10, and proxy10 has a connection 20 to server 12. Streams that are downloaded from server 12 may be routed over connection 20 to proxy10 before being passed to user 14, allowing the streaming content to be buffered and

smoothed at the proxy10. When user 16 later seeks to download content, a separate connection to server 12 can be established to allow for the download. In the case where user 16 is requesting the same content as is being viewed by user 14, user 16 a separate connection to server 12 may be unnecessary. User 16 can open a connection 22 to proxy 10, and the content can be downloaded directly therefrom (e.g., if proxy 10 has previously stored the content). By reducing the volume of data being downloaded from server 12 in this fashion, the overall network congestion is reduced and fewer overall packet losses may be experienced.

Now turning to **Figure 4**, one possible implementation of a proxy 10 is illustrated. It should be appreciated that this illustration does not show all of the components that may be needed to allow operation of the proxy in a real network. Rather, the focus is on the functional components that may be used to accomplish a data smoothing operation.

Shown in the figure is a case where an input stream 30 from a content source is applied to a receive buffer 32. Receive buffer 32 may, in practice, be a shared memory operated under the control of a memory controller that processes the incoming streams so as to store data packets thereof in one or more logical queues (which may themselves be implemented as linked lists of buffers). Thus, the data packets that make up the input streams are stored in a fashion that allows their respective sequence number (or other identifying criteria) and stream/connection to be identified.

Sequencer 34, which may be a general or special purpose processor and/or a custom integrated circuit configured to carry out the smoothing operations described herein, is responsible for assembling the smoothed stream within long-term storage unit 36 and or transmit buffer 38. That is, sequencer 34 is responsible for collecting and transferring to the long-term storage unit 36 (which again may be a shared memory and/or a linked list of buffers or another computer-readable medium such as a CD-ROM, etc.), the packets that will make up the smoothed outgoing stream(s) 40. Packets of a smoothed stream may be played out of transmit buffer 38 at a rate optimized for a receiving client under the control of sequencer 34 or a memory controller (not shown). Such a playback will occur under the conditions discussed above.

As has been indicated, the present smoothing scheme is particularly useful for stored media streams that utilize the keyframe/difference frame concept. When users request playback of data that does not coincide with a keyframe, the smoothing process will prevent a server from overwhelming a client with a burst of initial data. However, when the proxy has knowledge of the type of content being burst by the server, the present scheme becomes even more powerful.

For example, under conditions where the proxy knows what type of information is being downloaded, the proxy can act intelligently to reduce the amount of data that is being transmitted by the server, because such data is merely redundant or otherwise unnecessary. To illustrate, consider that some media recording schemes (for audio information in particular) use keyframes to embed sound format descriptions. Individual difference frames between keyframes and the difference frame at the time requested by a user may not be necessary for playback (i.e., because all the information needed by the receiving client is contained in the keyframe). Thus, such intervening difference frames should not be passed on to the client, even though the server may transmit them anyway. An intelligent proxy can recognize that these intervening frames are unnecessary and eliminate them from playbacks to the client. This will reduce the size of the initial burst, making the smoothing process that much more efficient.

In other audio recording schemes, the data for many segments of time is spread out over multiple packets in multiple frames, allowing for graceful degradation in the face of packet loss. Thus, to accurately play back audio for a user requested time "t", several frames worth of information may need to be downloaded from the server storing the requested content. Here, the smoothing process is again useful in as much as the initial download from the server will likely be a large burst of data.

In the video domain, similar characteristics may exist for various video encoding schemes. For example, the encoding scheme promulgated by the Moving Picture Experts Group (MPEG) defines several different types of difference frames, not all of which may be necessary to reconstruct a particular image requested by a user. Hence, the intelligent proxy, knowing that an MPEG movie has been requested can make decisions about what frames are

needed and what frames are not needed (i.e., from a burst provided by a server), and reduce the output stream to the client accordingly, to provide enhanced smoothing.

Thus a scheme for smoothing the download of streaming content broadcast over a public network or network of networks has been described. Although the foregoing description and accompanying figures discuss and illustrate specific embodiments, it should be appreciated that the present invention is to be measured only in terms of the claims that follow.

CLAIMS

What is claimed is:

1. A method, comprising smoothing a burst of information included in a data stream received from a content source across one or more computer networks at a proxy disposed between the content source and one or more content consumers so as to provide one or more output data streams to the one or more content consumers that reproduce the information included in the burst over a time interval longer than that in which it was received at the proxy.

2. The method of claim 1 wherein the proxy is disposed at the last media link between the content source and at least one of the content consumers.

3. The method of claim 1 wherein the data stream comprises content transported over the one or more computer networks according to an unreliable transport protocol.

4. The method of claim 3 wherein the transport protocol is real time transport protocol (RTP)

5. The method of claim 1 wherein the output data streams are delayed from the data stream received from the content source.

6. A proxy, comprising a sequencer configured to smooth a burst of information included in a data streams received from a content source across one or more computer networks so as to provide one or more output data streams to one or more content consumers that reproduce the information included in the burst over a time interval longer than that in which it was received at the proxy.

7. The proxy of claim 6 wherein the sequencer is configured to operate with a data stream transported over the one or more computer networks according to an unreliable transport protocol.

8. The proxy of claim 7 wherein the transport protocol is real time transport protocol (RTP).

9. The proxy of claim 6 wherein the proxy includes a cache.

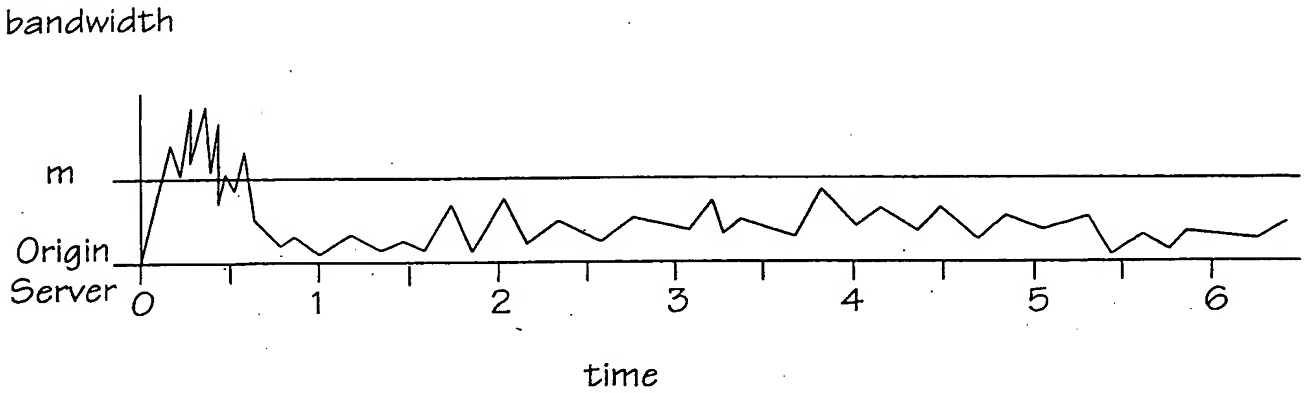

bandwidth
m
Origin Server
0
1
2
3
4
5
6
time

FIG. 1

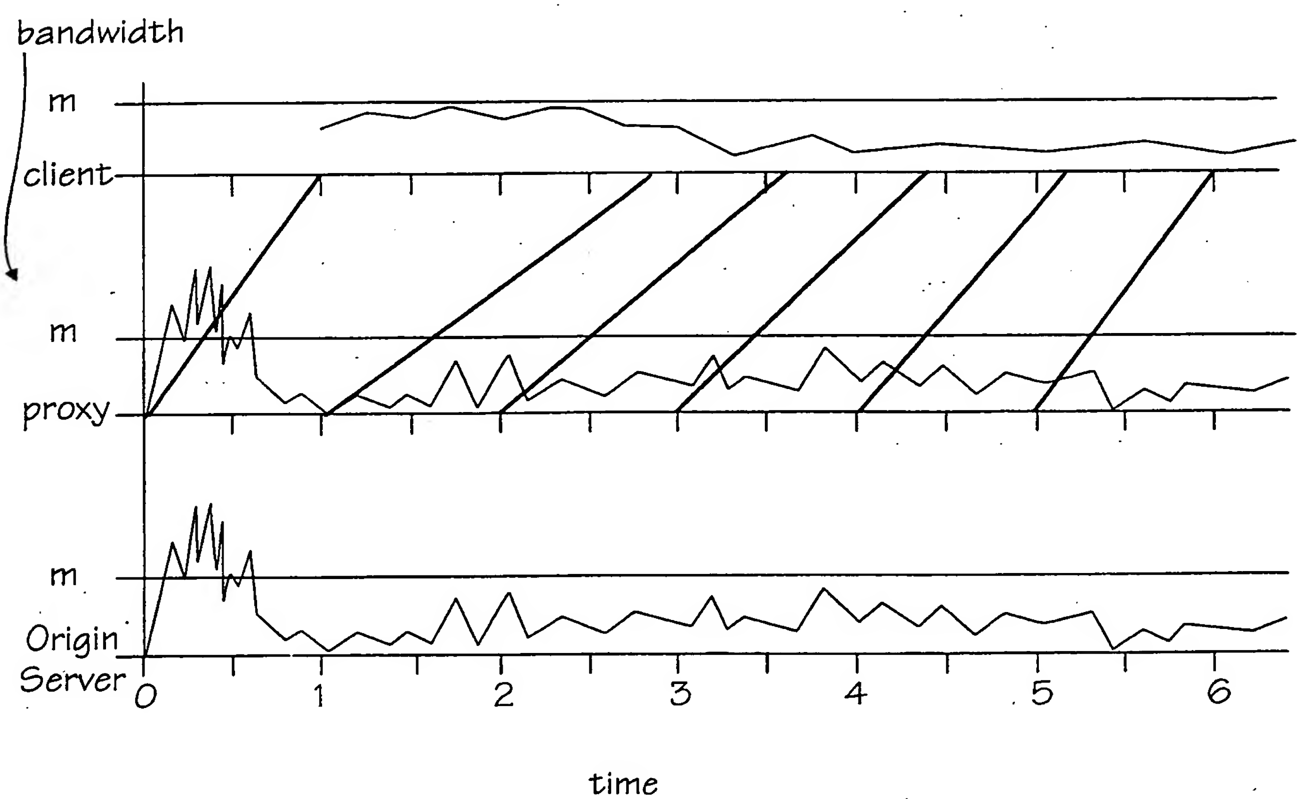

bandwidth
m
client
m
proxy
m
Origin Server
0
1
2
3
4
5
6
time

FIG. 2

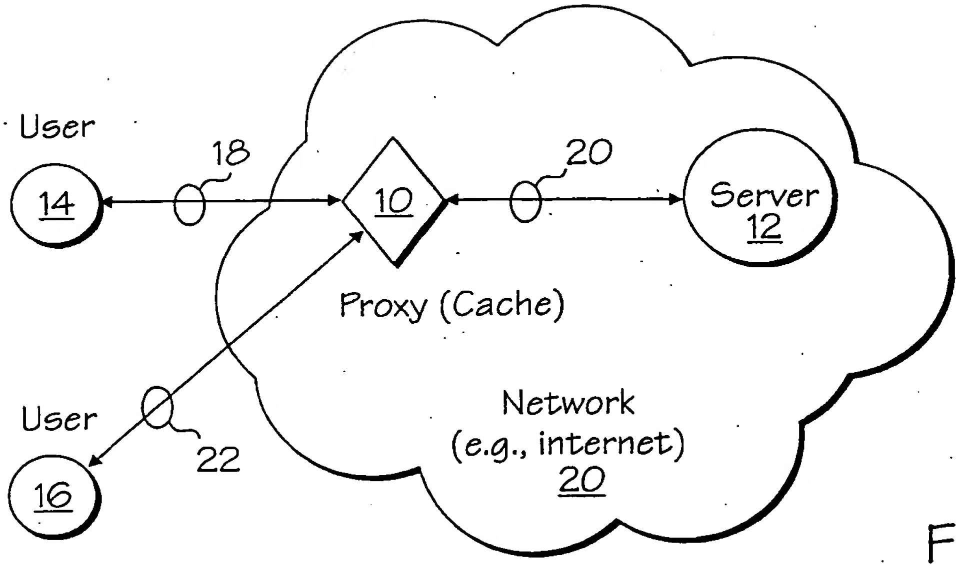

User
14
18
10
20
Server
12
Proxy (Cache)
User
16
22
Network
(e.g., internet)
20

FIG. 3

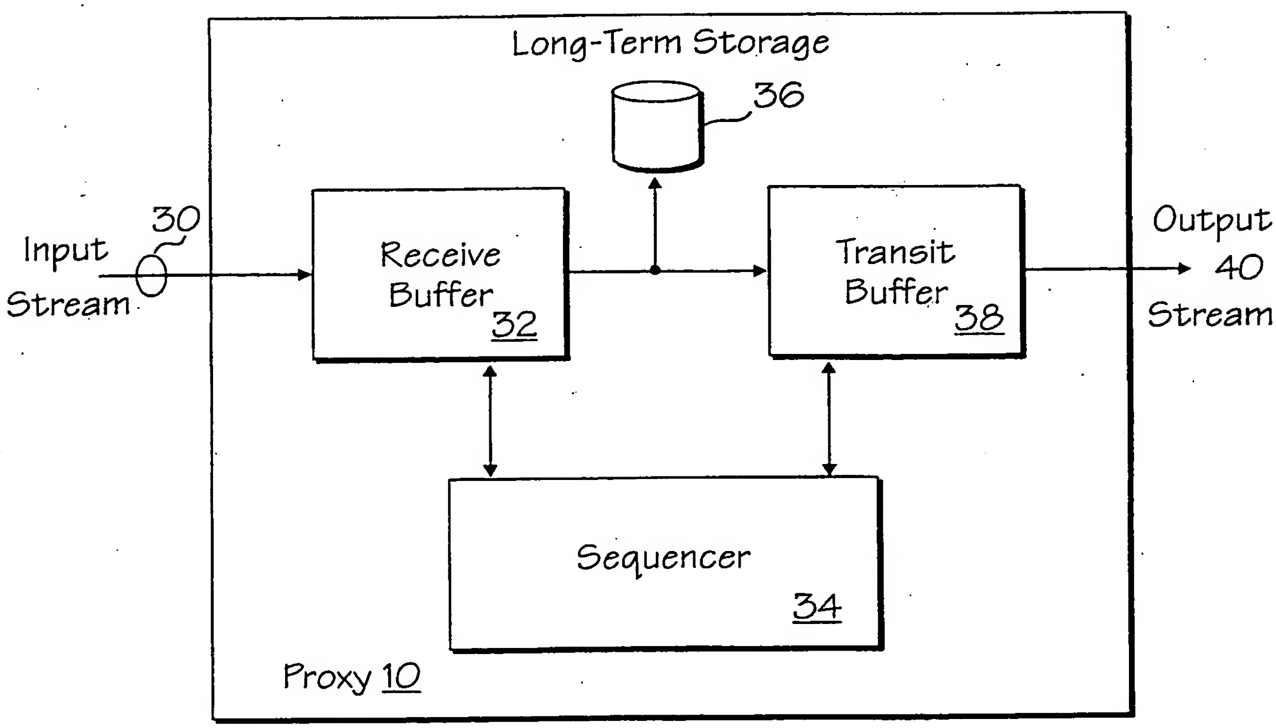

Long-Term Storage
36
Input
Stream
30
Receive
Buffer
32
Transit
Buffer
38
Output
40
Stream
Sequencer
34
Proxy 10

FIG. 4

INTERNATIONAL SEARCH REPORT

Intern ...nal Application No
PCT/US 00/11081

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L12/56

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	REXFORD J ET AL: "SMOOTHING VARIABLE-BIT-RATE VIDEO IN AN INTERNETWORK" IEEE / ACM TRANSACTIONS ON NETWORKING, US, IEEE INC. NEW YORK, vol. 7, no. 2, April 1999 (1999-04), pages 202-215, XP000828687 ISSN: 1063-6692 page 202, left-hand column, line 1 -right-hand column, line 47 --- -/--	1,2,5,6 3,4,7-9

[X] Further documents are listed in the continuation of box C.

[ ] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
25 August 2000	06/09/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Brichau, G

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
Y	G. MAMAIS, M. MARKAKI, M.H. SHERIFF AND G. STASSINOPOULOS: "Evaluation of the Casner-Jacobson Algorithm for Compressing the RTP/UDP/IP Headers" IEEE, 1998, pages 543-546, XP002145714 National Technical University of Athens abstract page 543, left-hand column, line 1 - line 23 ---	3,4,7,8
Y	SUBHABRATA SEN, JENNIFER REXFORD AND DON TOWSLEY: "Proxy Prefix Caching for Multimedia Streams" IEEE, March 1999 (1999-03), pages 1310-1319, XP002145715 University of Massachusetts, AT&T Labs abstract ---	9
P,X	JENNIFER REXFORD, SUBHABRATA SEN AND ANDREA BASSO: "A Smooting Proxy Service for Variable-bit-rate Streaming Video" GLOBECOM '99, December 1999 (1999-12), pages 1823-1829, XP002145716 AT&T Labs and University of Massachusetts page 1823, left-hand column, line 1 -page 1824, left-hand column, line 37 page 1829, right-hand column, line 1 - line 8 -----	1-9